10-601 Recitation #1

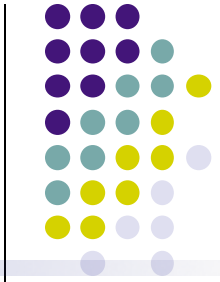
Function Approximation, Decision Trees, and Overfitting

September 14th, 2011

Shing-hon Lau

Office hours: Friday 3-4 PM

Agenda



- Administrivia
- Function Approximation
- Decision Trees
- Overfitting

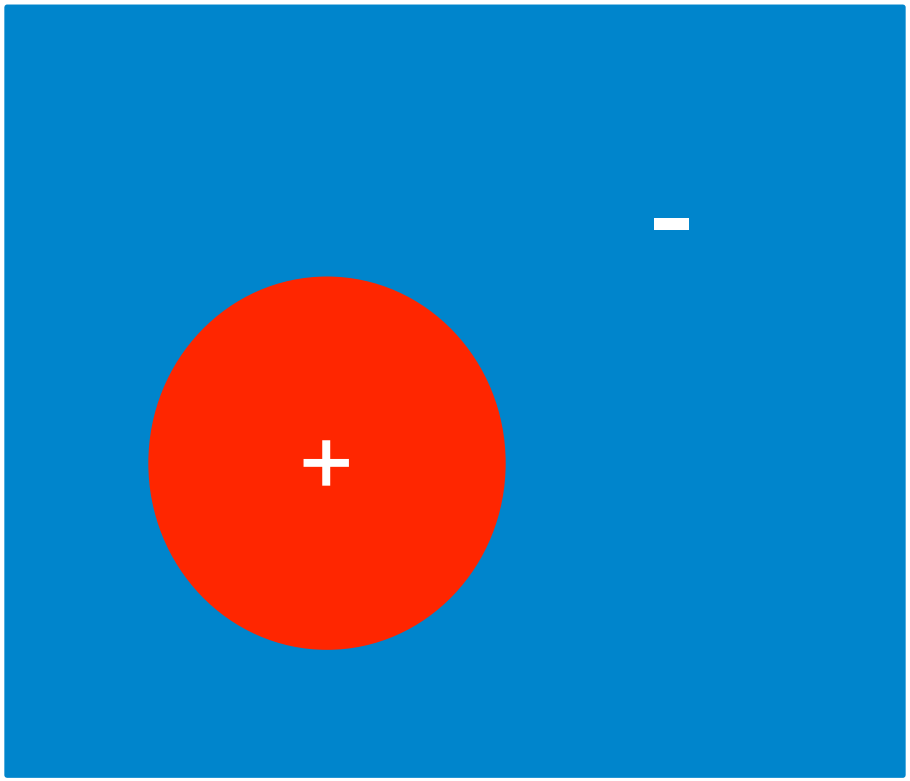
Administrivia

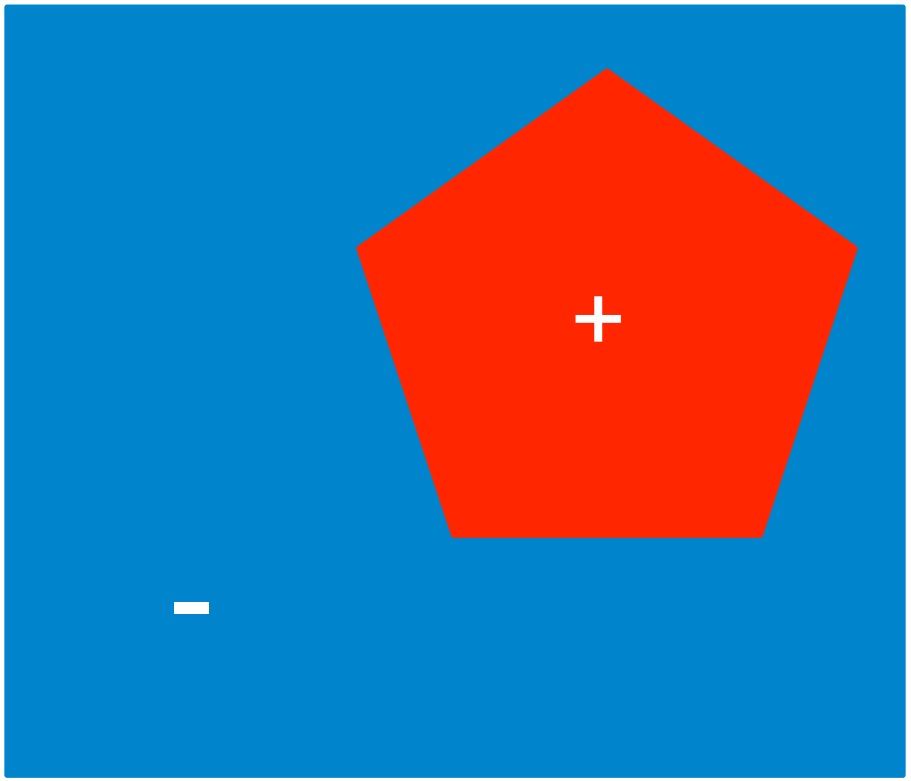
- Course website (www.cs.cmu.edu/~aarti/Class/10601/)
- Blackboard (www.cmu.edu/blackboard/)
- Homework #1 out soon (on BB + website)
- Slides online (on course website)
- Mailing lists (10601-instructors, 10601-announce)
- Recitation time/location

Function Approximation

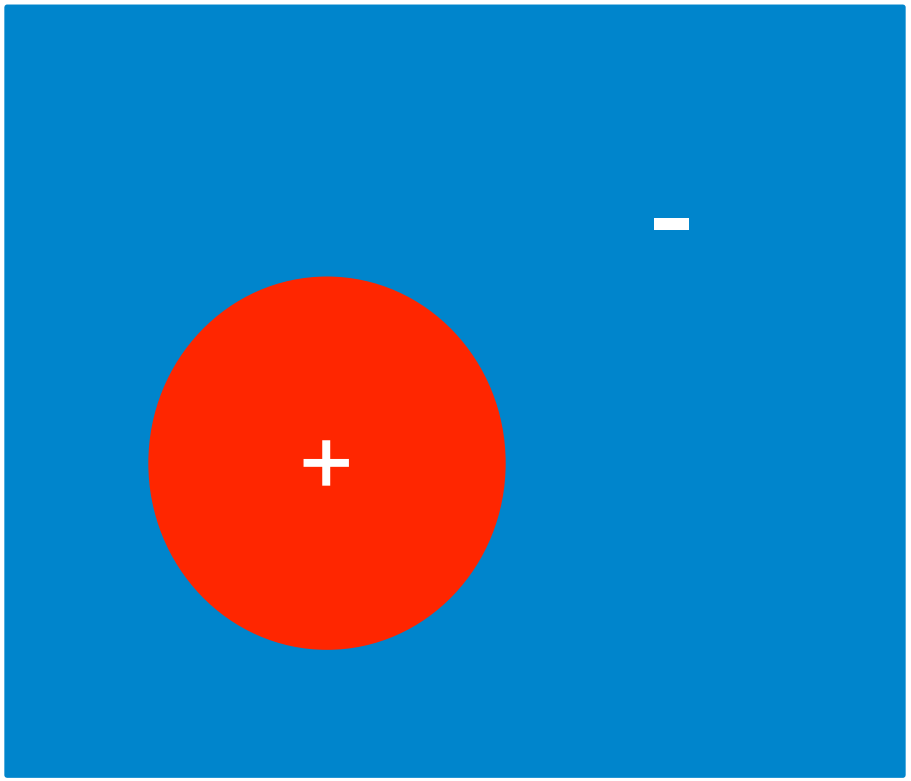
- Given:
  - Data X with labels Y
- Learn:
  - Function $f : X \rightarrow Y$
- Why can't we search through all possible functions?

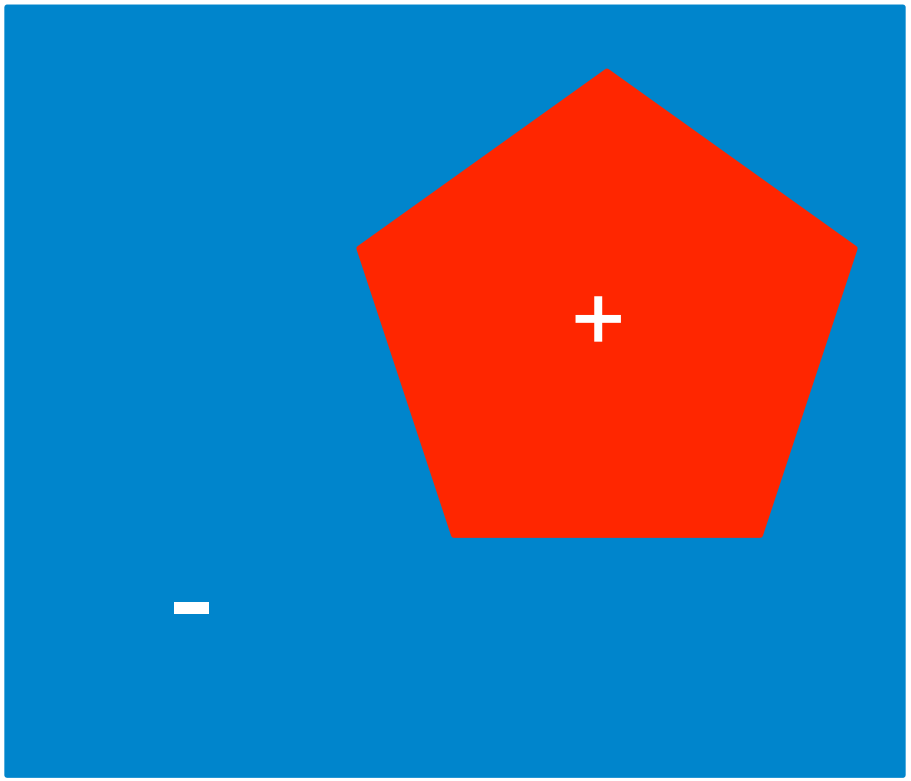
Function Approximation





Function Approximation



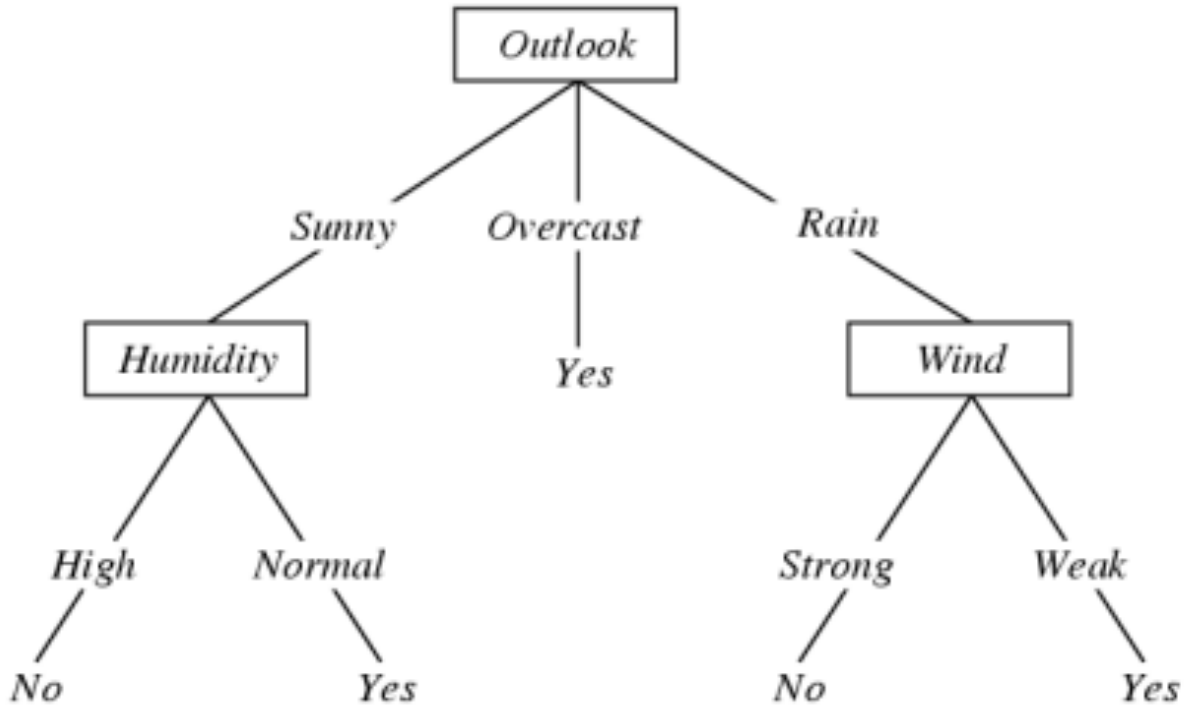


What does a Decision Tree look like?

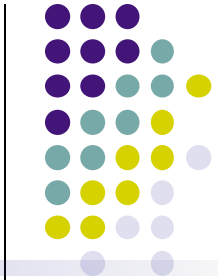
Decision Trees

A Decision tree for

F: <Outlook, Humidity, Wind, Temp> → PlayTennis?



Decision Tree Algorithm



Top-Down Induction of Decision Trees

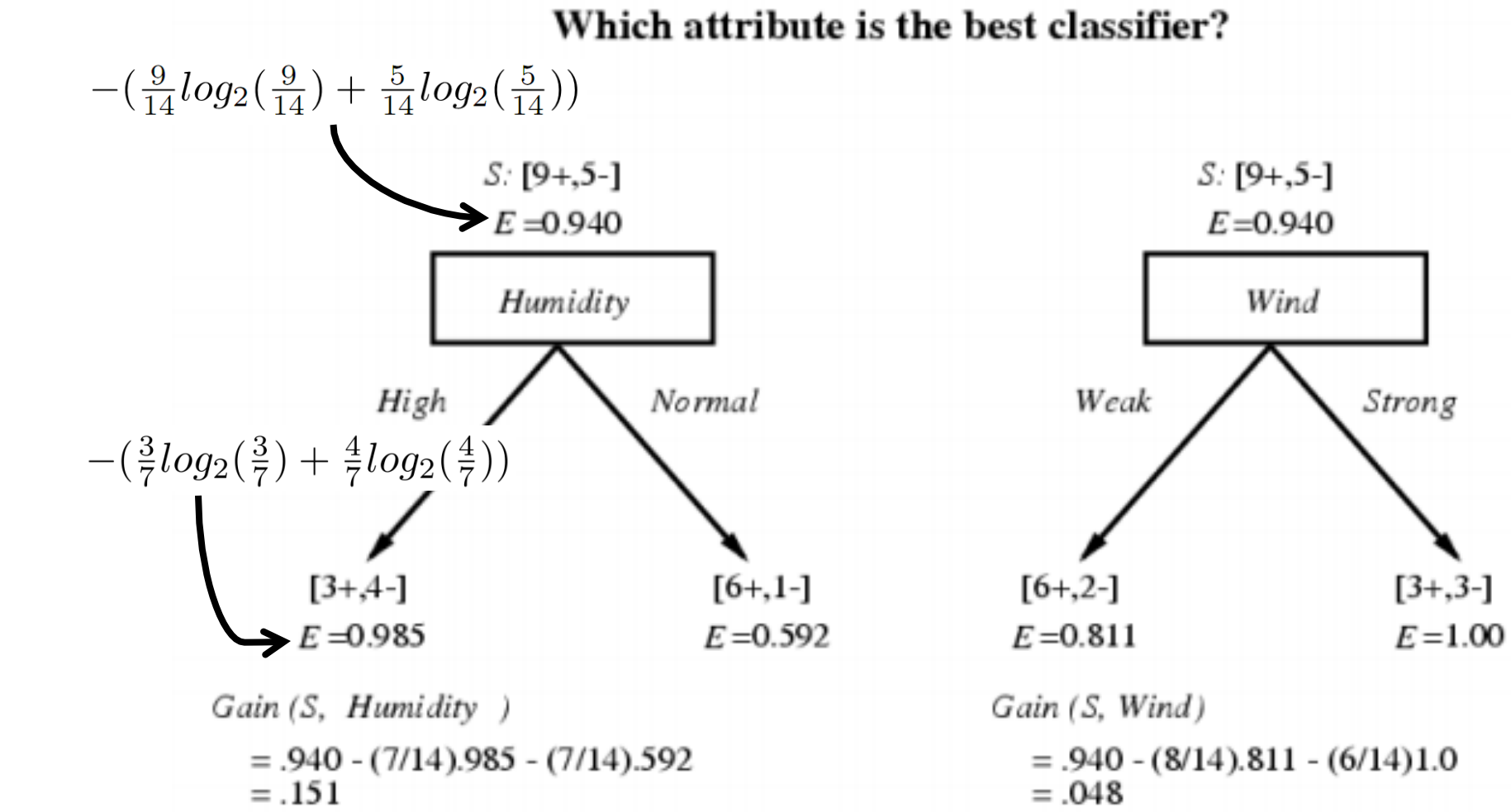
[ID3, C4.5, Quinlan]

node = Root

Main loop:

1. $A \leftarrow$ the "best" decision attribute for next *node*
2. Assign A as decision attribute for *node*
3. For each value of A, create new descendant of *node*
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

Defining "Best"

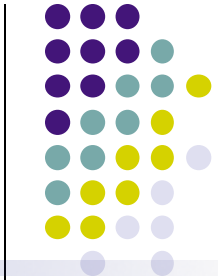


Continuous Variables

- Why can't we use the same algorithm we used for discrete variables?

- What can we do instead?

Decision Tree Algorithm



Top-Down Induction of Decision Trees

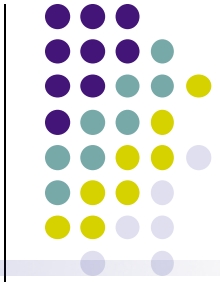
[ID3, C4.5, Quinlan]

node = Root

Main loop:

1. $A \leftarrow$ the "best" decision attribute for next *node*
2. Assign A as decision attribute for *node*
3. For each value of A, create new descendant of *node*
4. Sort training examples to leaf nodes
5. If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

Continuous Variables



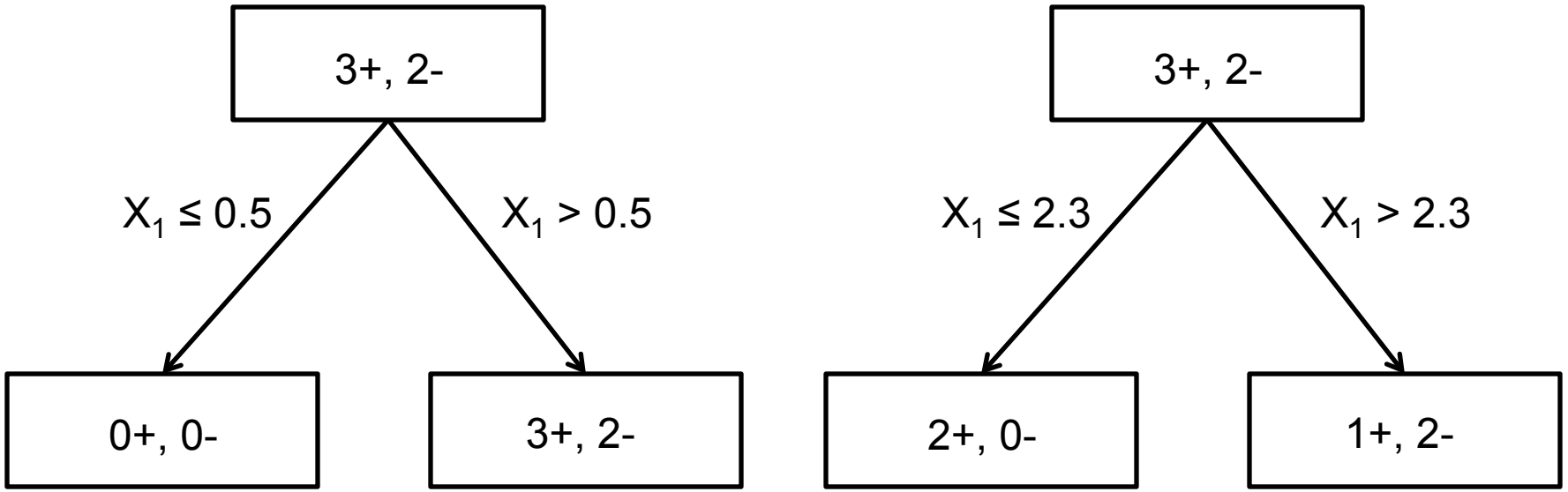
- Consider the following data:

X_1	1.0	2.0	2.6	3.4	4.4
Y	+	+	−	−	+

- No difference between $x_1 \leq 1.5$ and $x_1 \leq 1.51$ in terms of the split
- Just consider thresholds between data points

An Example

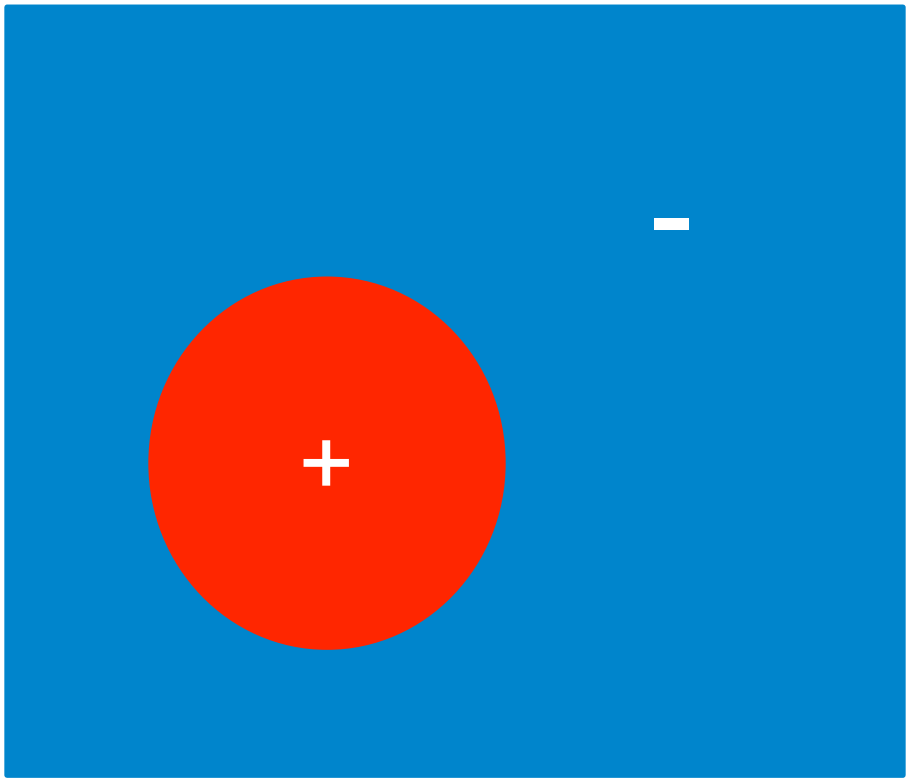
X_1	1.0	2.0	2.6	3.4	4.4
Y	+	+	−	−	+

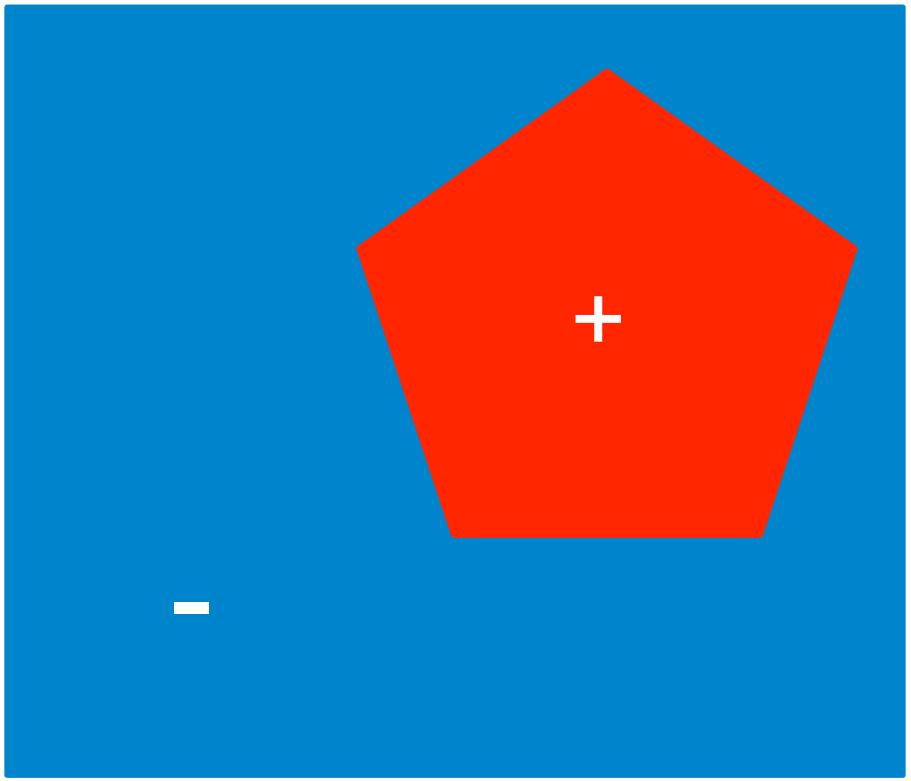


$$IG = 0.97 - 0.97 = 0$$

$$IG = 0.97 - \tfrac{2}{5}(0) - \tfrac{3}{5}(-(\tfrac{1}{3}log_2(\tfrac{1}{3}) + \tfrac{2}{3}log_2(\tfrac{2}{3}))) = 0.42$$

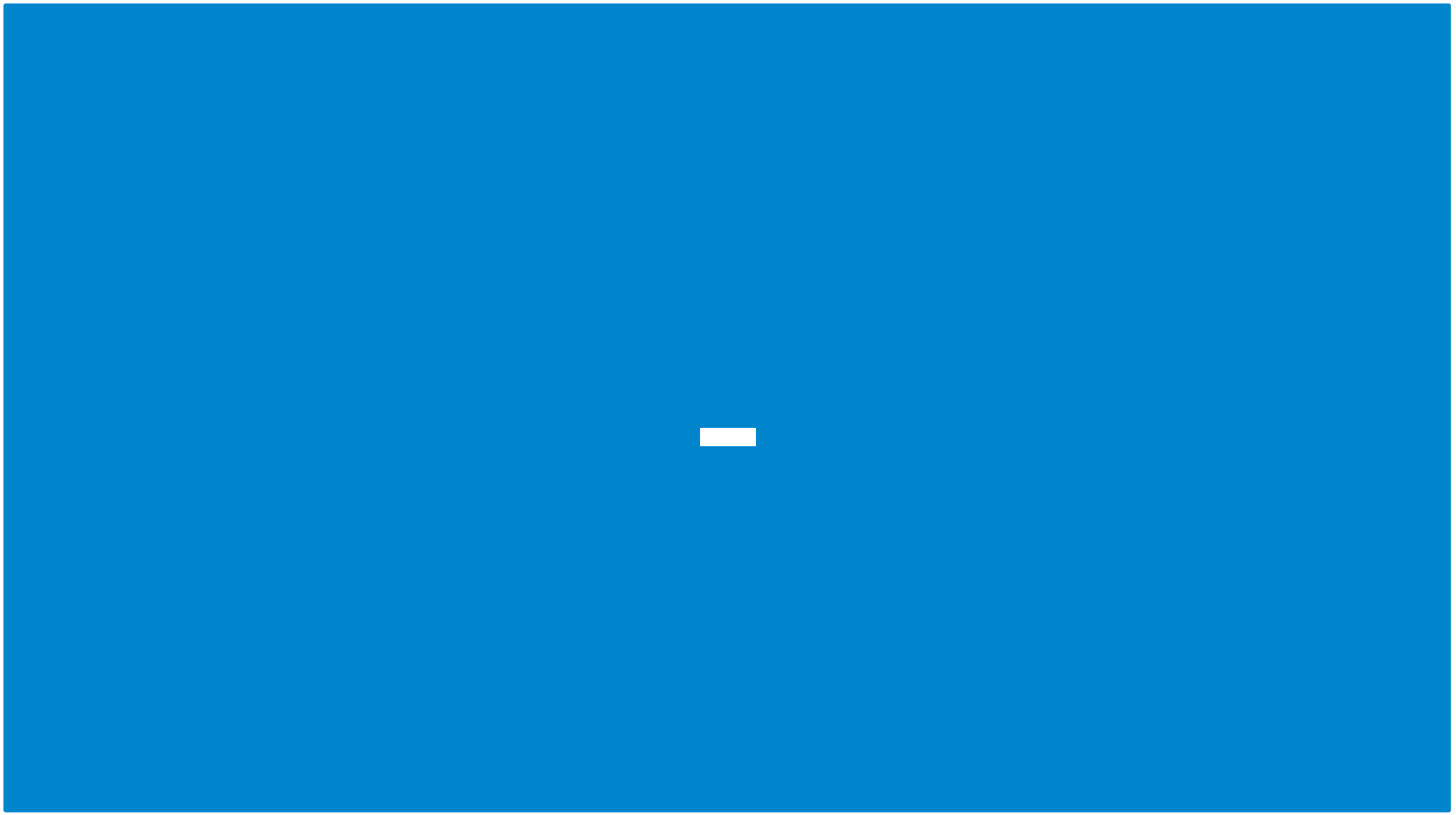
Function Approximation



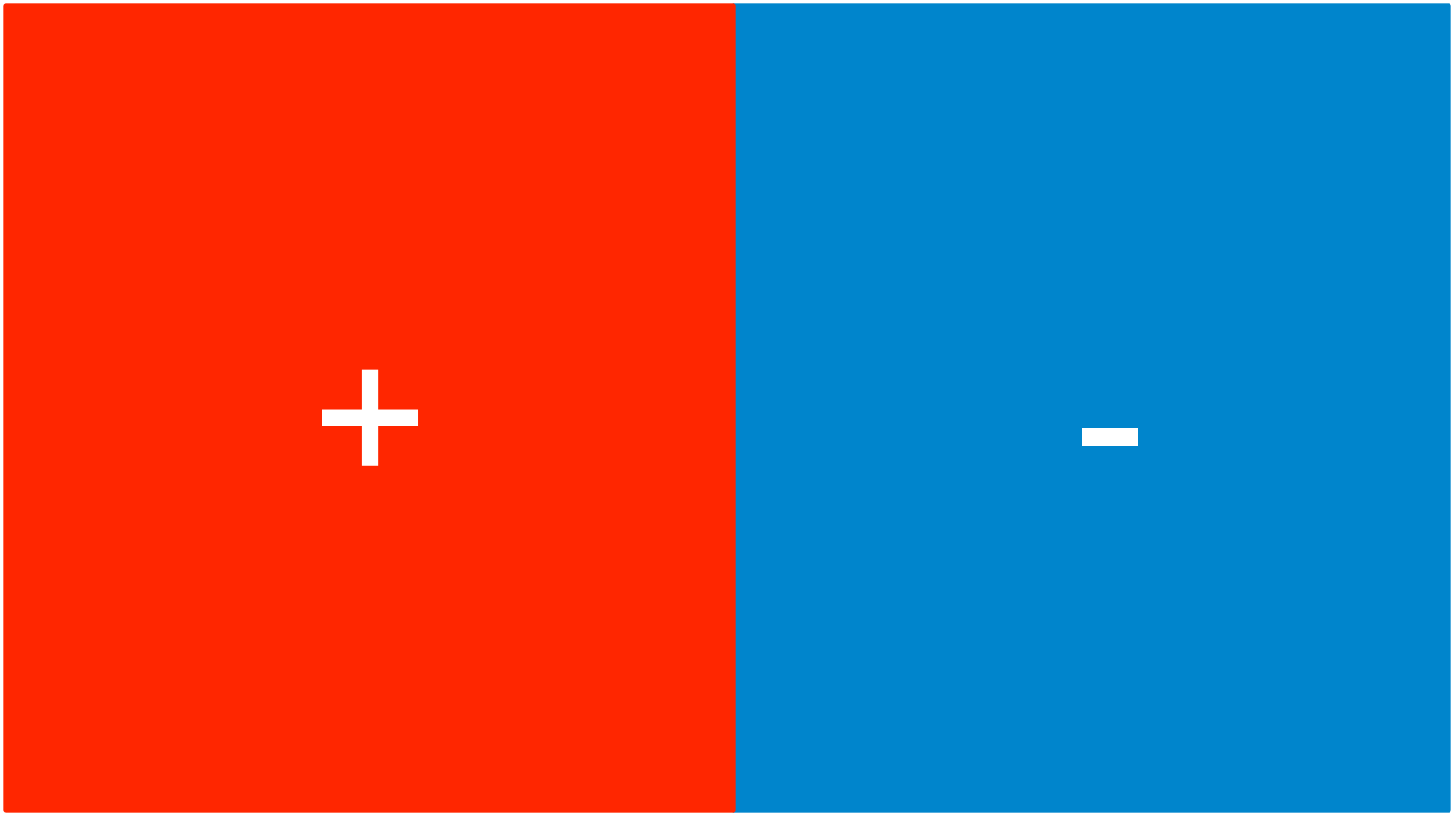


What does a Decision Tree look like?

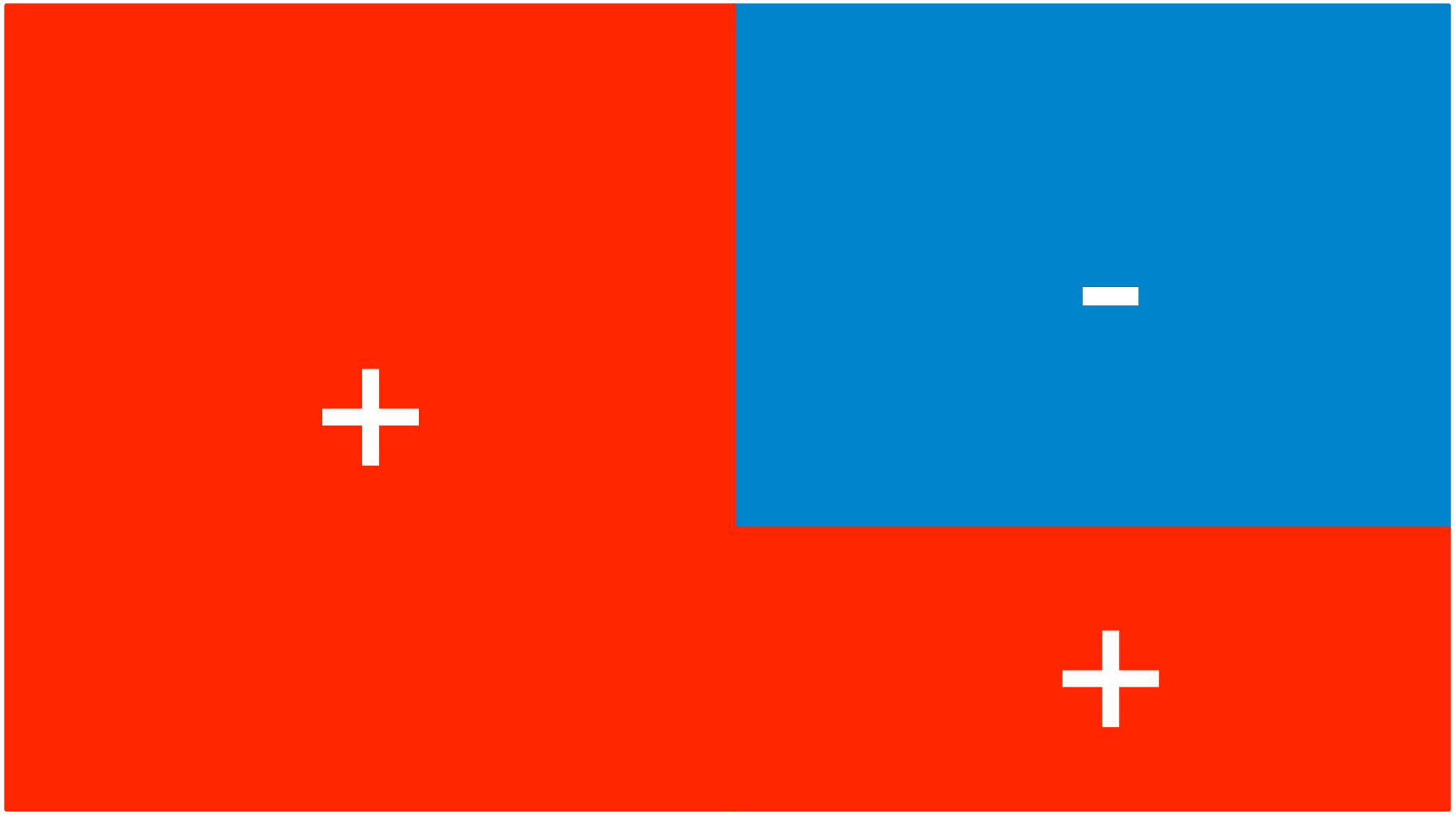
Function Approximation



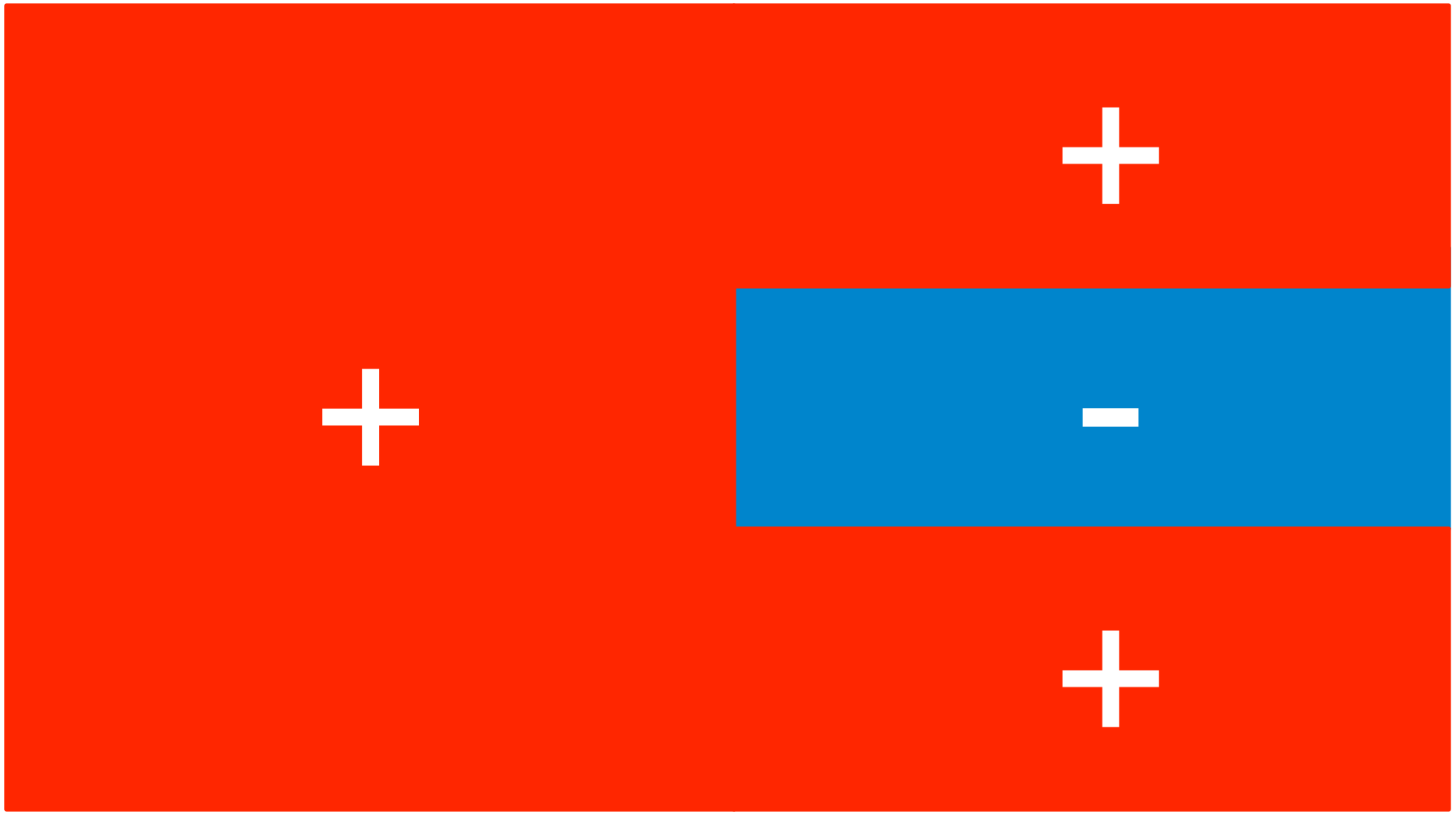
Function Approximation



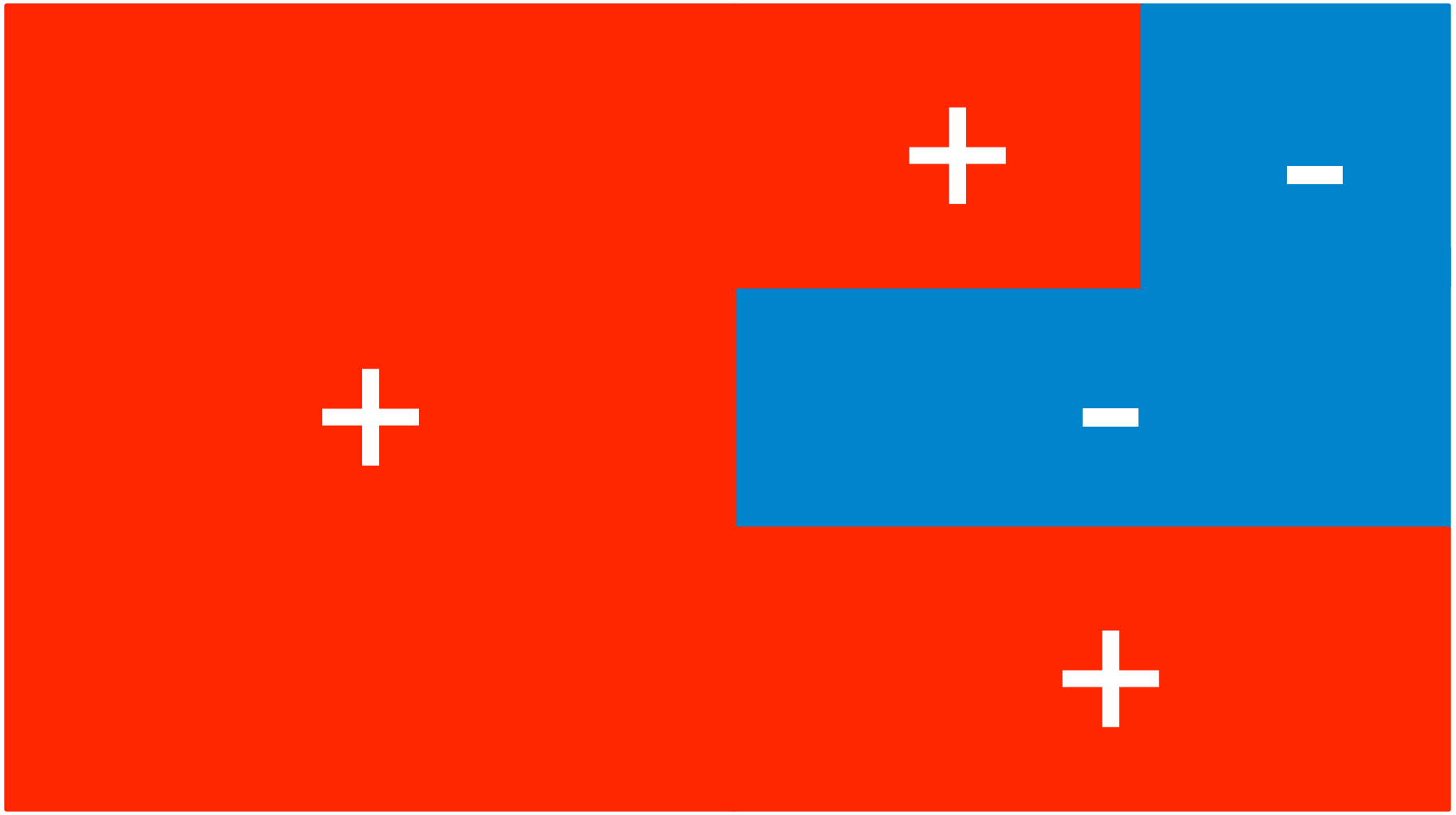
Function Approximation



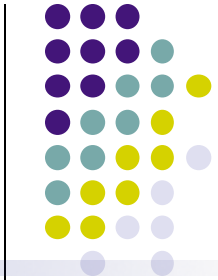
Function Approximation



Function Approximation



Overfitting



Consider error of hypothesis h over

- training data: $error_{train}(h)$
- entire distribution $\mathcal{D}$ of data: $error_{\mathcal{D}}(h)$

Hypothesis $h \in H$ **overfits** training data if there is an alternative hypothesis $h' \in H$ such that

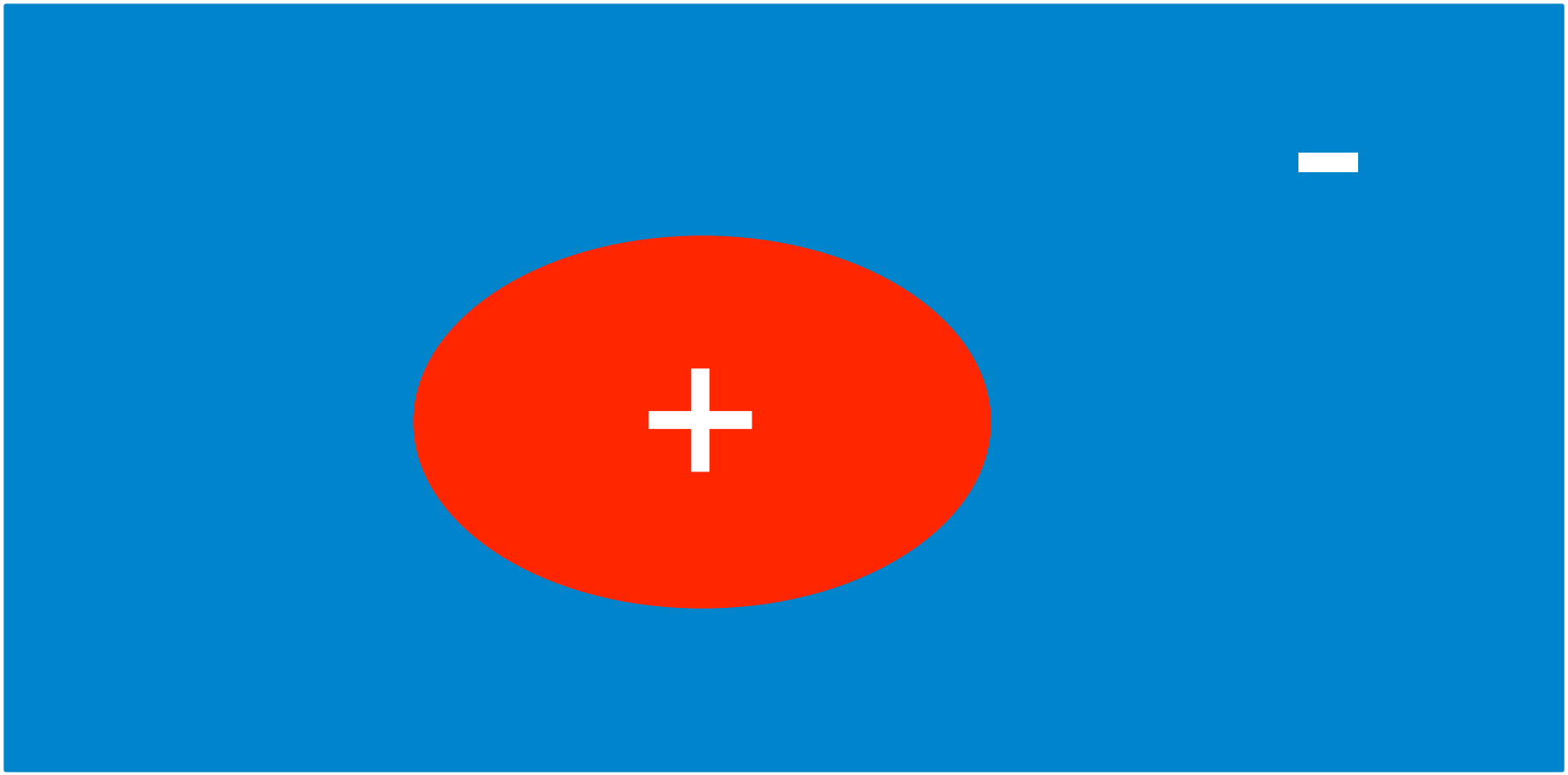
$$error_{train}(h) < error_{train}(h')$$

and

$$error_{\mathcal{D}}(h) > error_{\mathcal{D}}(h')$$

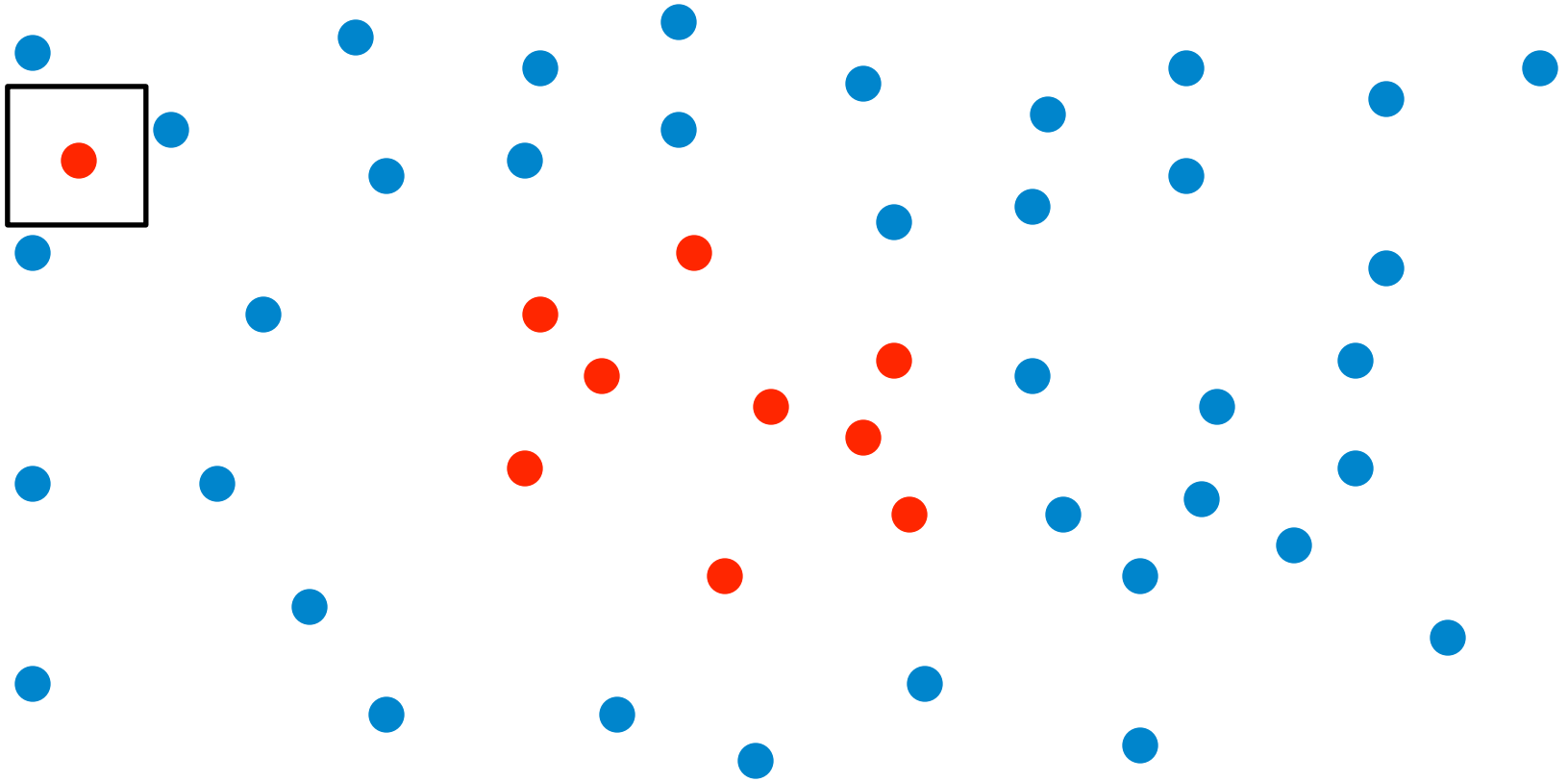
Overfitting in Pictures

- Suppose this true decision boundary:

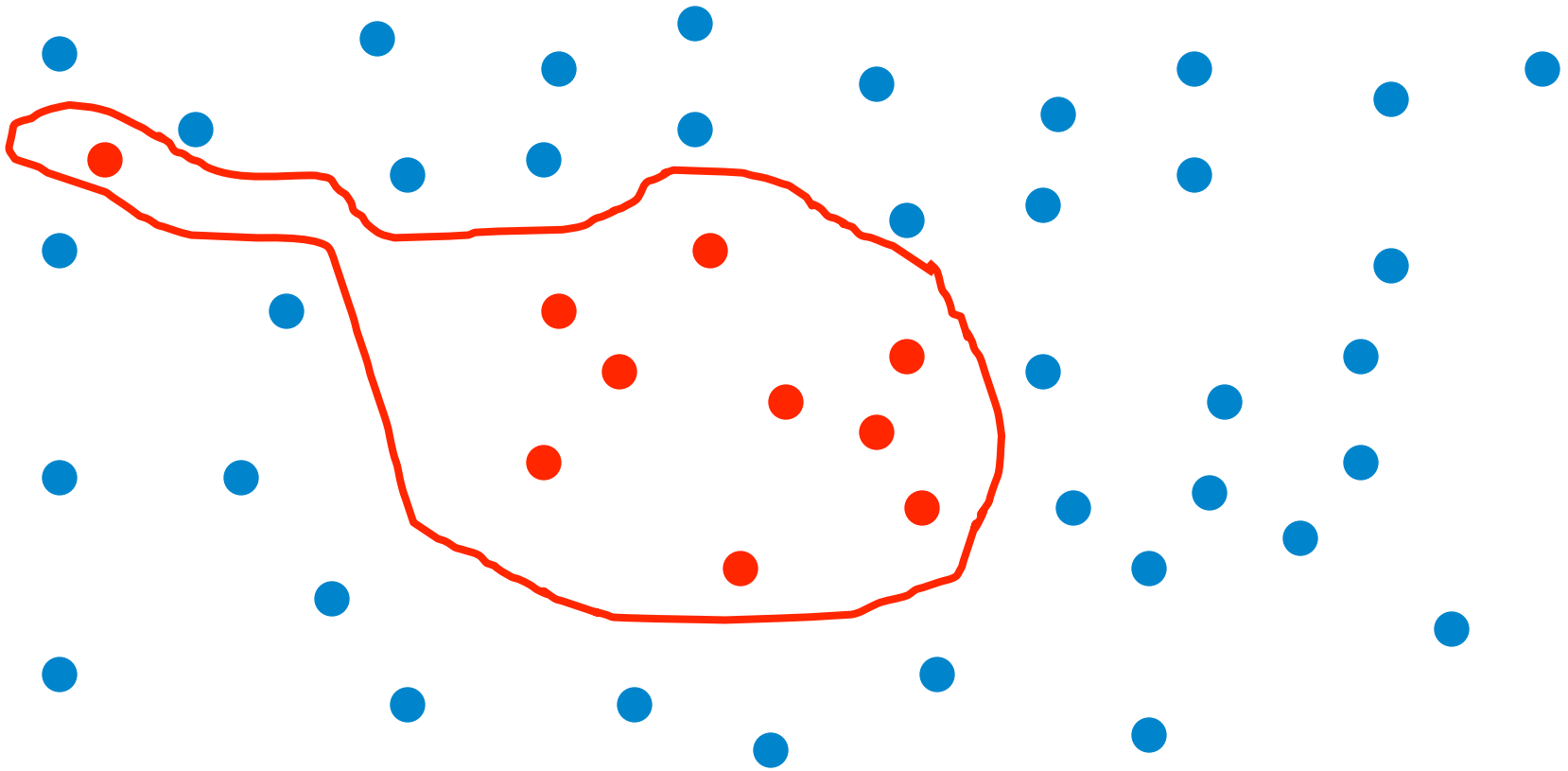


Overfitting in Pictures

- Gather these data points:

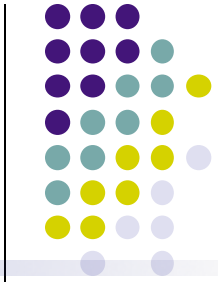


Overfitting in Pictures



- Learn this classifier:

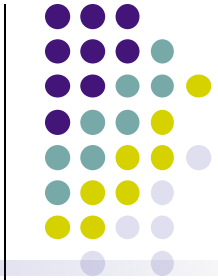
Avoiding Overfitting In Decision Trees



- Stop growing tree when splits are not statistically significant

- Grow a full tree and then do post-pruning

XOR Problem



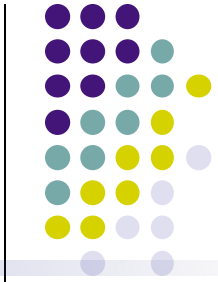
- Suppose we have this data:

X_1	X_2	Y
0	0	-
1	0	+
0	1	+
1	1	-

x 1000

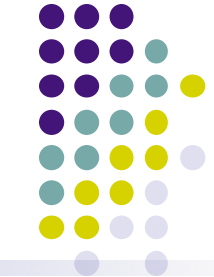
- No split is statistically significant, but two splits gives perfect classification

Post-pruning



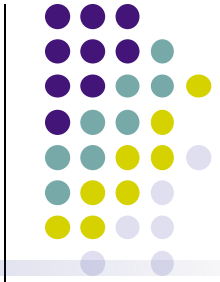
- Build full Decision Tree
- Consider pruning each leaf node
- Remove leaf node if it does not appreciably hurt validation error rate
- Repeat

Training, Validation, Testing



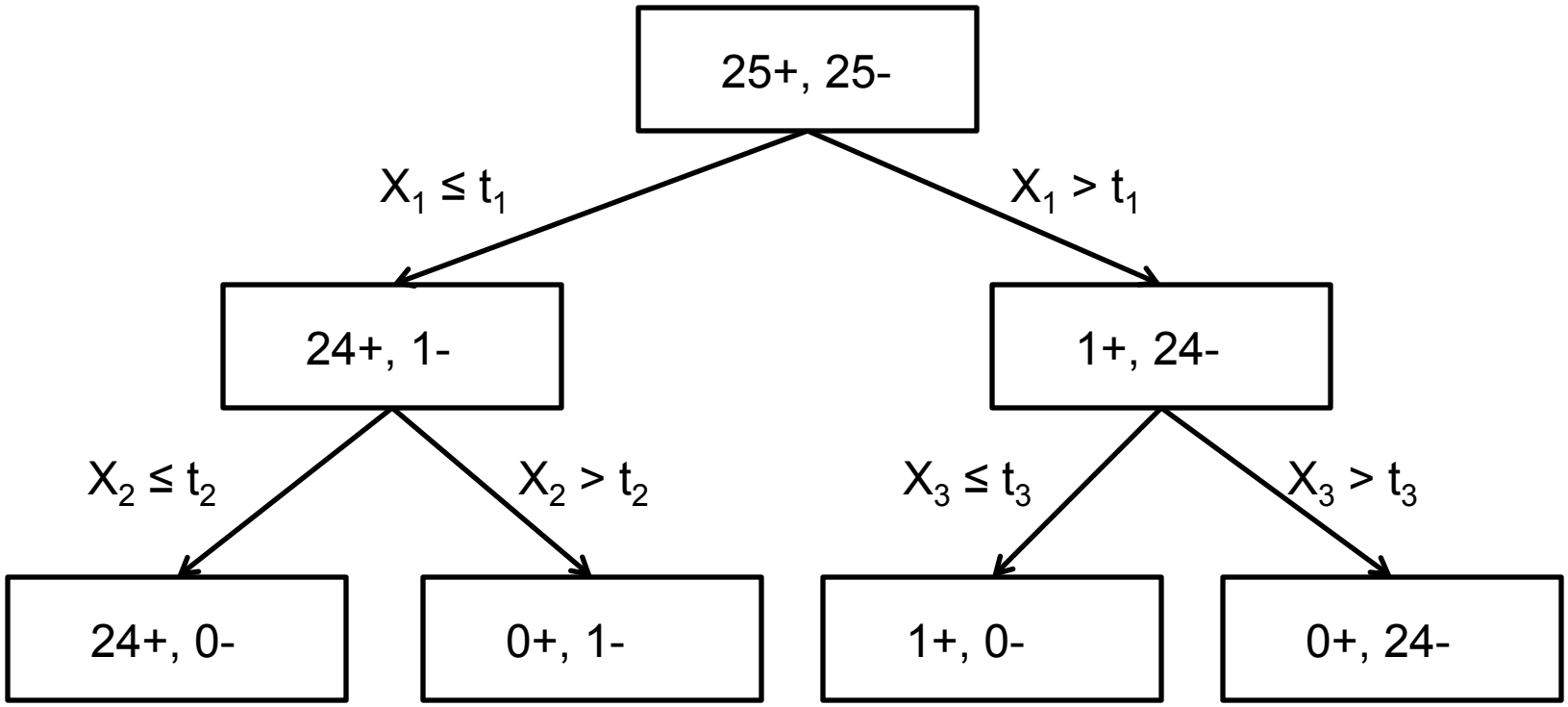
- Training set used to fit the classifier
- Validation set used to determine the goodness of the classifier
- Test set results are final outcome

Post-pruning



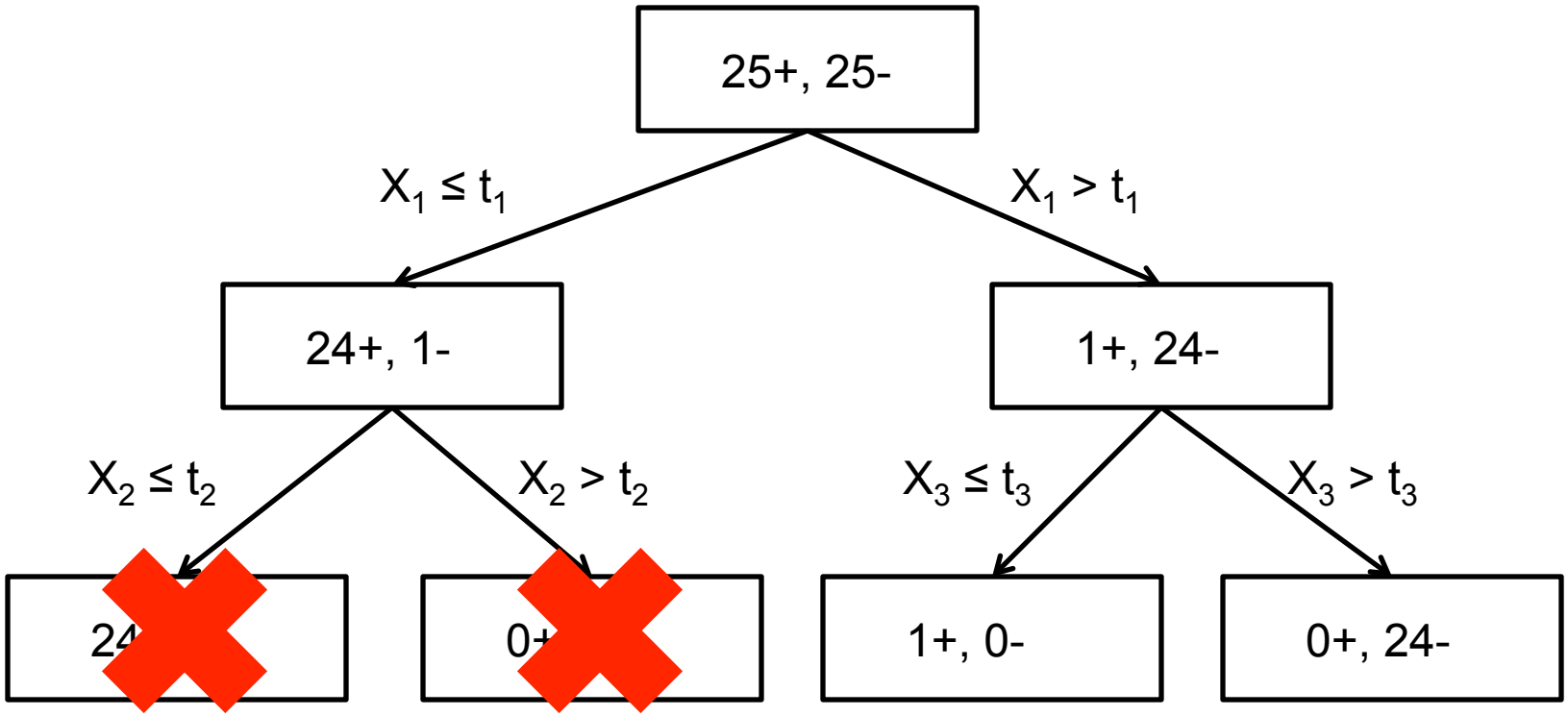
- Build full Decision Tree
- Consider pruning each split
- Remove split if it does not hurt validation error rate
- Repeat (in a greedy and iterative fashion)

Post-pruning



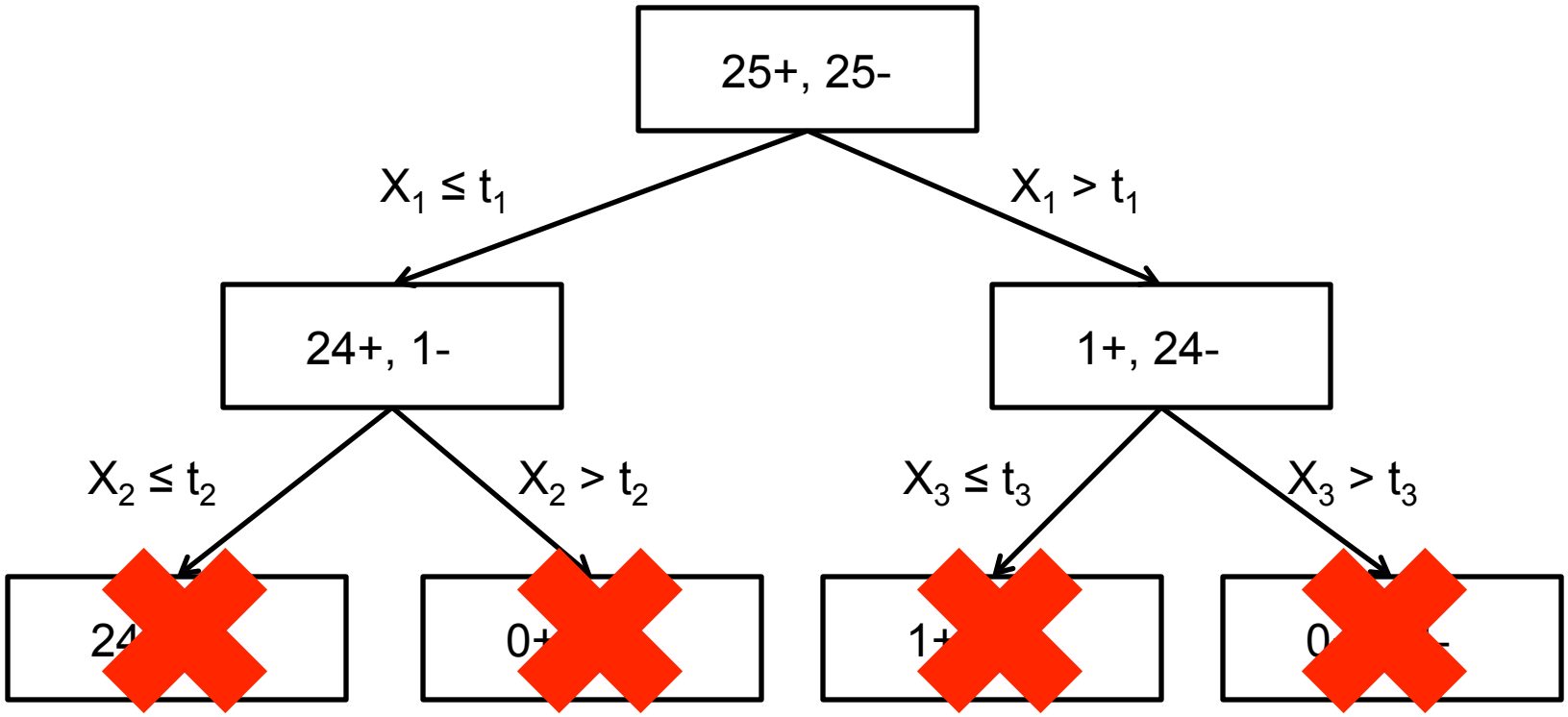
Validation Set Accuracy: 80%

Post-pruning



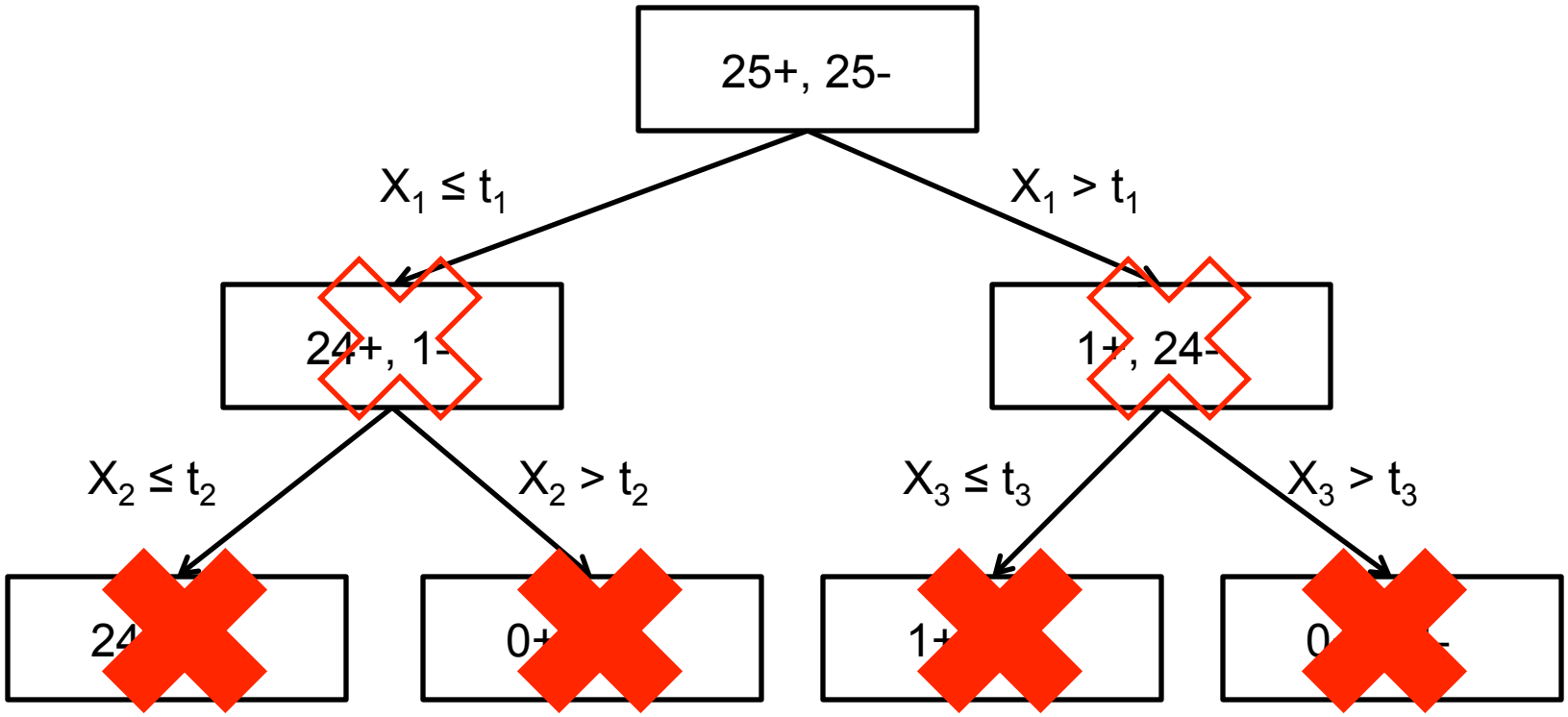
Validation Set Accuracy: 85%

Post-pruning



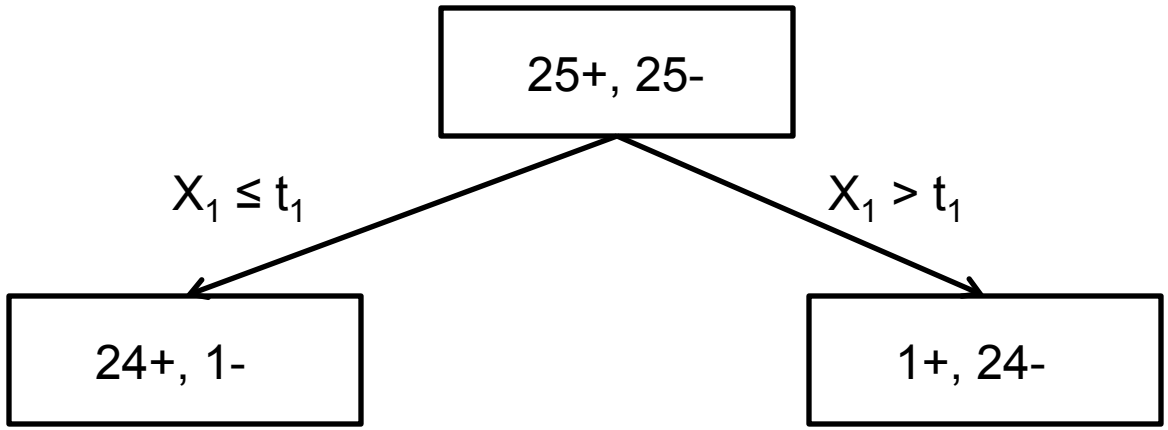
Validation Set Accuracy: 90%

Post-pruning



Validation Set Accuracy: 50%

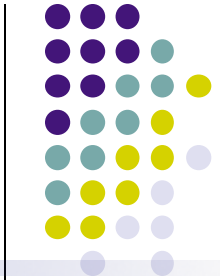
Post-pruning



Final Decision Tree

Extra Slides

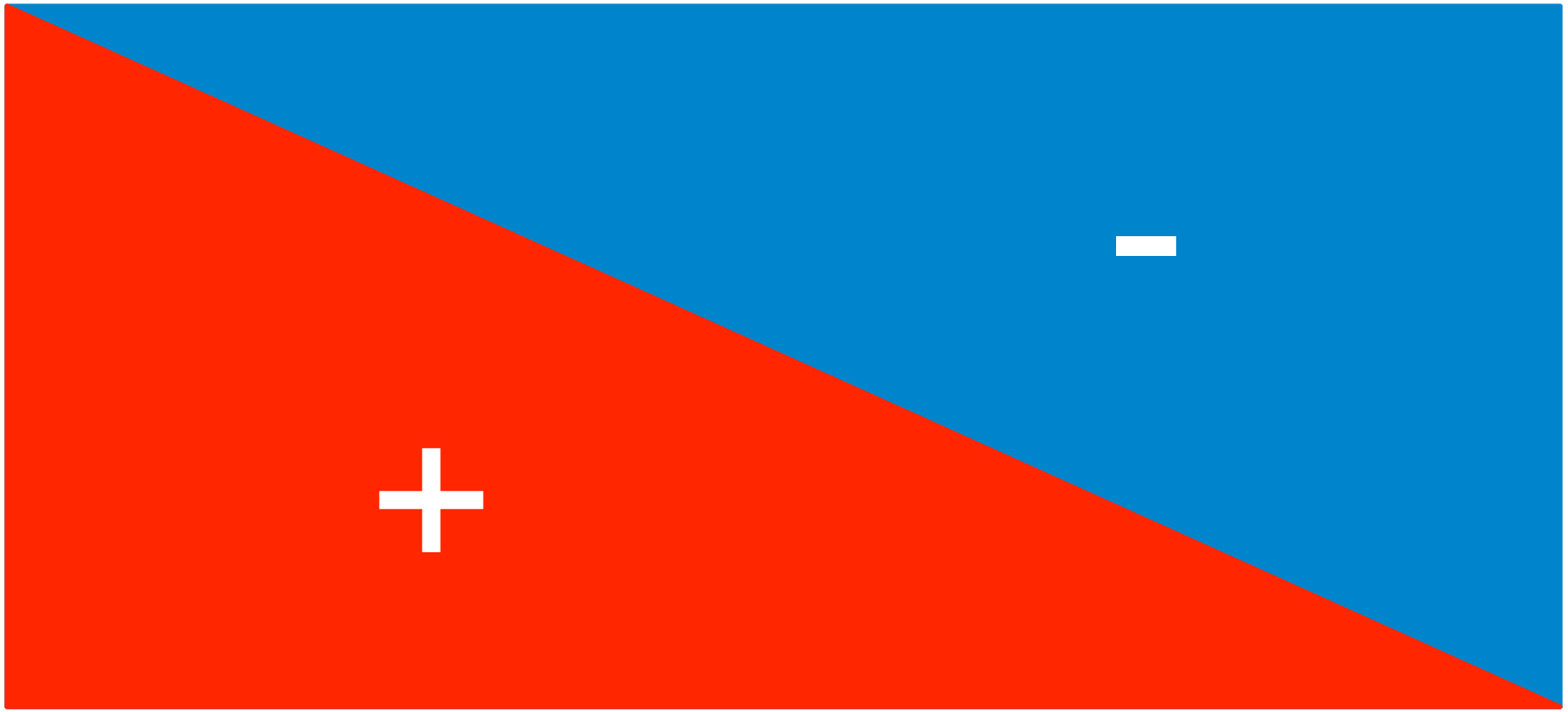
Variability of Decision Trees



- Slight changes in the data can significantly affect what Decision Tree is learned

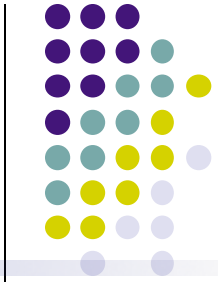
- Changes in splits near the top of the tree will completely change subsequent splits

Capturing Additivity



How can we capture this structure in a DT?

Extensions



- Regression Trees
- Handling missing values

- Bagging, Boosting (e.g., Adaboost), Random Forest